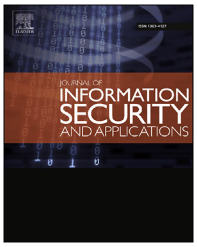
Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa



Trust-based privacy-aware participant selection in social participatory sensing

Haleh Amintoosi [b,a,*], *Salil S. Kanhere* [a], *Mohammad Allahbakhsh* [c,a]

[a] *The University of New South Wales, Sydney, Australia*
[b] *Ferdowsi University of Mashhad, Mashhad, Iran*
[c] *University of Zabol, Zabol, Iran*

ARTICLE INFO

Article history:
Available online 11 November 2014

Keywords:
Social participatory sensing
Trust
Privacy
Collusion detection

ABSTRACT

The main idea behind social participatory sensing is to leverage social friends to participate in mobile sensing tasks. A main challenge, however, is the identification and recruitment of sufficient number of well-suited participants. This becomes especially more challenging for large-scale online social networks with unknown network topology and complex friendship relations. Moreover, the potential sparseness of the friendship network may result in insufficient participation, thus reducing the validity of the obtained information. In this paper, we propose a participant selection framework to address the aforementioned limitations. The nomination module of the framework makes use of a customised random surfer to crawl the requester's social graph and identify suitable nominees. The nominee selection is determined as a function of the members' suitability scores and pairwise trust perception among members. The selection module is responsible for selecting the required participants from the set of nominees based on the nominee's timeliness, the number of participants selected so far and the task's remaining time. Moreover, the selection is done in a way that prevents from the formation of a colluding group among the selected participants. Simulation results demonstrate the efficacy of our proposed framework in terms of selecting a large number of reputable participants with high suitability scores, in comparison with state-of-the-art methods.

1. Introduction

1.1. *Background*

Web 2.0-enabled applications harness the wisdom of crowds and human intelligence to collaborate and accomplish a wide variety of tasks such as creating content, performing tasks, etc. (Murugesan, 2007; Doan et al., 2011). The recent improvements in mobile phone technology and sensing capabilities (such as microphone, camera, accelerometer, GPS, etc) in particular, has led to the emergence of a new exciting paradigm known as participatory sensing (Burke et al., 2006). In participatory sensing, the key idea is to recruit ordinary people to crowdsource data from their mobile phones (Kanhere, 2013).

The integration of participatory sensing systems with online social networks has resulted in the emergence of social

* *Corresponding author*. The University of New South Wales, Sydney, Australia.
E-mail addresses: haleha@cse.unsw.edu.au (H. Amintoosi), salilk@cse.unsw.edu.au (S.S. Kanhere), mallahbakhsh@cse.unsw.edu.au (M. Allahbakhsh).

http://dx.doi.org/10.1016/j.jisa.2014.10.003

participatory sensing (Krontiris and Freiling, 2010) in which, social network members can act as service requesters and utilise social friends and friends-of-friends as participants to contribute to their tasks. A pertinent example of such a system is Jelly[1] which is built on top of existing social networks like Facebook[2] and Twitter.[3] When the users encounter something unusual, they can take a picture of the object, formulate a query and submit it to their social network.

1.2. *Problem statement*

One of the main challenges in the success of social participatory networks is identifying and recruiting sufficient number of well-suited participants. Typically, there is no explicit incentive for participation and people contribute altruistically. In the absence of adequate contributors, there is a danger that the application will fail to gather meaningful data. Another challenge, particularly for tasks which require domain-specific knowledge is the suitability of the participants to collect appropriate data (Reddy et al., 2010). It is logical that contributions produced by suitable participants should be trusted more than those prepared by others. On the other hand, some participants might be biased or malicious. They may even build collusive groups in order to maximise their unfair benefits by supporting the requesters of their own interest. So, it is desirable to identify the colluders in order to prevent their recruitment. To sum up, identifying and recruiting sufficient reputable and well-suited participants is essential for the reliability of the task outcome.

Existing solutions for identifying and recruiting participants in social participatory networks such as (Ehrlich et al., 2007; Alkouz et al., 2011; Zhang et al., 2007) mostly suffer from several limitations. First, they are based on the assumption that the social graph topology and the social links between users are known to the third-parties and applications. Although this may be true when utilising an organisational social network or a small social network (such as groups of acquaintances, co-workers, alumnae, etc.), it is not realistic in large-scale social networks such as Facebook or Twitter with hundreds of millions of users.

The second limitation is the potential sparseness of a requester's friendship graph which makes sufficient recruitment even more challenging. The requester may have few friends or may lack close friends who may be willing to contribute. It has been shown that online social networks can be considered as sparse (Ugander et al., 2011). For example in Yahoo! Pulse[4] which is an online social network involving hundreds of millions of users, almost half of the users only have one friend connection (Yang et al., 2011).

The third limitation is the inefficiency of existing recruitment techniques and their vulnerability against malicious activities. Generally, the participant recruitment is done through following three methods: (i) the open call (ii) the auction and (iii) pre-selection (Surowiecki, 2005; Vukovic and Bartolini, 2010; Satzger et al., 2013) (More details about the existing recruitment methods are presented in Section 4). The shortcoming of the open call method is its unsuitability for the tasks with pre-defined quality requirements. In the auction method, the responsibility of assessing the contributions' quality and fidelity is on the requester, which may prove to be an exhausting task (Chandler et al., 2013). The main problem with the pre-selection method (which is also relevant for the other two methods) is its vulnerability to collusion. A group of malicious participants might form a colluding group and sway the outcome of the task in accordance with their agenda.

1.3. *Contributions and outline*

In this paper, we propose a participant recruitment framework for social participatory sensing. Contrary to our previous work (Amintoosi and Kanhere, 2013a, 2014, 2013b, in press), we assume that the social graph topology is not known to our framework. The main implication of this assumption is that there is no prior access to the profile information of social network members, and hence, the identification and selection of suitable participants must be done on-the-fly while the social graph is being traversed. We also assume that the graph search is not uniform and not limited to a specified depth. Consequently, the graph search may progress deeper in some friendship chains and shallower in others, depending on the social trusts along the chain. We believe that making these assumptions will make our proposed framework more practical and closer to the reality.

The proposed framework consists of two main modules, which aim at addressing the key challenges raised above. The first module, known as the *nomination module*, is responsible for identifying well-suited members (known as *nominees*) in the requester's social graph and inviting them to contribute. We define the nominees as those who (i) are suitable to contribute, and (ii) there exists a trustworthy path to them (starting from the requester). We argue that in order to have a comprehensive view of the participant's suitability, the following parameters should be taken into account: (i) the participant's expertise (in order to satisfy the task requirements), (ii) his reputation score (as an indication of being a highly trustable participant), (iii) the pairwise privacy score between the requester and participant (to minimise the privacy breach of requester's sensitive information), (iv) the requester's list of preferred participants (to give priority to those who are preferred by the requester to be recruited), and (v) the requester's blocked list (those with whom, the requester is reluctant to contribute (more details in Section 2.1.1). As mentioned above, we assume that the structure of the entire social network is not accessible. Hence, the nomination module relies on a customised random surfer to crawl the social network graph (originating at the requester) and identify well-suited nominees. The number of initiated random surfers for a requester is equal to the number of his friends. Each random surfer starts from one of the requester's friends and chooses its next visited nodes based on the suitability score of the nodes as well as the pairwise trusts along the path. Once the nominees are identified, they are invited to attend in the task.

The *selection module*, the second module of our recruitment framework, is responsible for selecting the final participants

[1] http://blog.jelly.co/post/72563498393/introducing-jelly.
[2] http://facebook.com.
[3] http://twitter.com.
[4] pulse.yahoo.com.

from the set of nominees who have accepted the invitation. In this module, we propose a time-aware and collusion-free recruitment method which is an extension to the pre-selection recruitment approach and aims at addressing the collusion issue. The selection module takes into account a set of parameters and decides whether to select the nominee. These parameters are (i) the selection score (i.e., the ratio of participants selected so far to the total number of required participants), (ii) the remaining time to the task deadline and (iii) the timeliness of the participant in previous tasks. The intuition behind considering these parameters is that when the task's remaining time is short, it is logical to select the nominee that has shown timely behaviour in his past contributions, as he is most likely to submit his contribution before the imminent deadline. If eligible to be selected, a final check is done to insure that the selection of this participant will not result in potential collusion. In particular, we aim to identify whether the addition of each new participant to the previously selected group will result in the formation of a group of colluders. Several indicators are used in literature to detect collusive behaviours (Allahbakhsh et al., 2013; Mukherjee et al.; Lim et al., 2010). We use the (i) group size (i.e., number of colluders), (ii) group support count (i.e., number of tasks in which colluders have collaborated in the past), and (iii) group time window (i.e., the time difference between the latest and earliest contribution of group members) as the indicators. The intuition behind considering these parameters is that the colluders usually form a group which is large enough to create a considerable impact. Moreover, group members usually target a substantial number of tasks and collaborate together in contributing to these tasks. The colluders also desire to contribute during a short time window (groups working over a long time window are unlikely to have worked together). By considering all these indicators, we determine the collusion possibility for each eligible participant by utilising the well-known Frequent Itemset Mining technique (Grahne and Zhu, 2005).

In summary, the main contributions of the paper are as follows:

- We propose a suitability assessment technique to compute the suitability score of a participant based on his expertise, reputation score, the pairwise privacy score between the requester and participant and the requester's list of preferred and blocked list (more details in Section 2.1.1).
- To identify the suitable participants, we propose a customised random surfer inspired by the idea of random walks and Markov chain. The proposed random surfer while provides the system with a set of suitable nominees, aims at addressing the bootstrapping problem for the newly-joined social network members by giving them the equal chance of being nominated as more reputable participants (more details in Section 2.1.2).
- We propose a time-aware collusion-free selection module, responsible for selecting the participants from the set of invited nominees. To do so, the participant's timeliness, the task deadline and the selection score are considered. A collusion possibility is also calculated for each eligible participant to prevent any possible collusion (more details in Section 2.2).
- The accuracy and usability of the proposed techniques has been tested via simulations using real-world datasets from the Advogato social network, Wikipedia Adminship Election and StackOverflow.[5] The evaluation results show superiority of our method over common recruitment methods.

The remainder of the paper is organised as follows. In Section 2, the proposed framework is explained in detail. Section 3 discusses the evaluation scenarios and results. The related works are investigated in Section 4. Finally, Section 5 concludes the paper and presents future directions.

2. Participant selection framework

The participant selection framework proposed in this paper consists of two main modules: *nomination* and *selection*. In this section, we first begin with the nomination module in Section 2.1 and will explain the selection module in Section 2.2.

2.1. *Nomination module*

A participatory task or simply a *task* can be is represented by θ_i, with its owner known as the *requester*. When defining the task, the requester may wish to define a set of requirements. For instance, he may require members who have a specified level of reputation (in terms of the history of previous contributions), or who live in a particular geographical region or who possess certain specific expertise (Reddy et al., 2010). These requirements act as criteria for the identification and selection of eligible participants. In order to identify eligible participants, the nomination module is responsible for crawling the social network graph and identifying potential candidates whose profile information match the task requirements. These candidates form the nominated group and will be invited to contribute to the task.

As explained in Section 1.2, constraining the graph crawling only to friends may lead to insufficient nominees, due to the potential sparseness of friendship graph or lack of enough experts among friends. Therefore, in our proposed method, we extend the social graph crawling deeper in the social graph in order to maximise the possibility of finding well-suited participants. In our framework, the nomination module relies on a customised random surfer to find well-suited participants. In the following, we first explain the evaluation of member's suitability score in Section 2.1.1 and then, describe the customised random surfer in detail in Section 2.1.2.

2.1.1. *Suitability assessment*

In order to evaluate the suitability of a member, a set of parameters should be considered and evaluated. In the following, we first explain the evaluation of each parameter in detail and then discuss the calculation of suitability score.

2.1.1.1. *Reputation*. The requester may specify a minimum level of the reputation as a requirement for participation, in order to obtain high quality contributions. We assume that a

[5] stackoverflow.com.

reputation management system such as (Amintoosi and Kanhere, 2014; Huang et al., 2012) is already in place, which calculates a reputation score ρ_i for each member m_i, where $m_i \in M$ and M is the set of social network members ($M = \{m_i 1 \le i \le n\}$). We also assume that the required reputation score of the task is denoted by ρ_{req}. In that case, the required level of reputation score for m_i to participate in task θ_j, denoted by ρ_i^j is as follows:

$$\rho_i^j = \begin{cases} \rho_i & \text{if } \rho_i \ge \rho_{req} \\ 0 & \text{otherwise} \end{cases}$$

The higher the value of ρ_i^j, the higher the eligibility of the participant to contribute to the task. We assume that ρ_i^j is a number in the range of [0,1].

2.1.1.2. *Expertise*. Expertise is defined as the measure of a participant's knowledge and is particularly important in tasks that require specific knowledge about a particular domain such as programming skills, familiarity to a geographical area, proficiency with a particular language or so on. Greater credence is placed in contributions made by a participant who has expertise in the task. Expert finding systems such as (Ehrlich et al., 2007; Alkouz et al., 2011) may be employed for evaluating expertise that analyse explicit (e.g., public profile data and group memberships) as well as implicit information (e.g., textual posts) to extract user interests and fields of expertise (Alkouz et al., 2011). We denote the level of match between the *i*th member's expertise and the *j*th task requirements by $E_i^{\theta_j}$. Assume that the E_j^t is the set of skills required by the task θ_j and E_i^m is the set of skills of the member m_i, then

$$E_i^{\theta_j} = \frac{\left|E_j^t \cap E_i^m\right|}{\left|E_j^t\right|}$$

$E_i^{\theta_j}$ is a number in the range of [0,1]. The higher the value of the $E_i^{\theta_j}$, the higher the match between the member's profile and the task requirement.

2.1.1.3. *Privacy requirements*. Privacy preservation has always been a great concern in social networks. When discussing about privacy, it is important to specify what defines failure to preserve privacy. One type of privacy breach occurs when a piece of sensitive information about an individual is disclosed to an adversary (Zheleva and Getoor, 2011). In social participatory sensing, privacy leakage may implicitly occur during the recruitment process through the nature of the query. In particular, it may lead to the disclosure of the requester's geographical location, his personal interests, political or religious views and so on. For example, if the requester asks for the vegetarian restaurants in a specific geographical area, it is probable that he is vegetarian and lives in that place. So, it is desirable to maximise the privacy preservation of the requester in the recruitment process.

There are different solutions for evaluating the pairwise privacy scores (some have been discussed in our previous work (Amintoosi and Kanhere, in press)). In this paper, we assume that the probability of a privacy breach in one-hop neighbourhood of the requestor is zero. This is reasonable since friends are assumed to be trustworthy. For other non-friend nodes, the probability of privacy breach is greater in nodes who have been involved in greater number of tasks initiated by that requester. The intuition behind this assumption is that the more m_i attends in tasks initiated by m_j, the more of m_j's sensitive information will be revealed to him. So, the pairwise privacy score will decrease as the number of mutual tasks increases. With this intuition in mind, the pairwise privacy score of giving (the non-friend) m_i the permission to contribute to the task θ_j initiated by m_j, denoted by $Pr_i^{\theta_j}$ is calculated via the following function:

$$Pr_i^{\theta_j} = \begin{cases} 1 - \left(\frac{t_{ij}}{T}\right)^2 & \text{if } t_{ij} \le T \\ 0 & \text{otherwise} \end{cases}$$

where, t_{ij} is the number of tasks m_i has done for m_j so far, and T is a system defined parameter which denotes the maximum number of the tasks initiated by m_j that m_i can participate in, following which m_i's privacy score reduces to zero.

2.1.1.4. *Requester's preferred and blocked lists*. The requester may be provided with a list of preferred participants, whom he prefers to recruit in his future tasks. This list may be automatically generated by the application (typically containing the requester's friends who have demonstrated trustworthy behaviour) or manually by the requester (based on his trust upon them). It is clear that those who appear in this list should be assigned a higher suitability score. So, for the member m_i who is being considered for nomination for the task initiated by m_j, we define the parameter Pf_{ij} to be 1 if m_i belongs to the m_j's preferred list, and zero otherwise. Similar to the requester's preferred list, a blocked list may also be available which contains the list of those whom the requester desires to exclude form the list of contributors due to their poor behaviour in previous tasks or privacy issues. Obviously, those belonging to this list should not be nominated. So, B_{ij} is set to 1 if m_i belongs to the m_j's blocked list, and zero otherwise.

2.1.1.5. *Computing the suitability score*. Once the above parameters are evaluated, they should be combined to arrive at a single value for the member's suitability score. To do so, the suitability score for a member m_i to attend in task theta$_j$ initiated by m_j, referred to as σ_i, is calculated as a weighted sum of parameters as:

$$\sigma_i = \begin{cases} 0 & \text{if } B_{ij} = 1 \\ w_1 \times \rho_i^j + w_2 \times E_i^{\theta_j} + w_3 \times Pr_i^{\theta_j} + w_4 \times Pf_{ij} & \text{otherwise} \end{cases}$$

where w_i is the weight of each parameter, and $\sum_{i=1}^4 (w_i) = 1$. The adjustment of the weights is application-dependant. For example, for privacy-aware applications, w_3 is set to be considerably high to give more importance to privacy parameter. Similarly, for tasks where expertise requirements are important, a higher weight may be associated with expertise (w_2). The suitability score is in the range of [0,1].

2.1.2. *Customised random surfer*

The participant selection framework proposed in this work is inspired by the concept of random surfer, well-known for its application in Google PageRank (Page et al., 1999). The main idea of random surfing is as follows. One of the graph nodes is selected randomly as the staring node, from which, the surfer starts its journey. The random surfer, then, picks one of the neighbouring nodes randomly and moves to that node. This process is repeated for a fixed period of time or till there is no outgoing link to go further. The random surfer concept is widely used for graph processing such as node ranking and clustering (to be further discussed in Section 4). Assume that for a node p_i, the number of outgoing links is denoted by $|p_i|$. Then, the stochastic matrix Π representing the random surfer is defined is as follows:

$$\Pi_{n\times n} = \begin{pmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{pmatrix}$$

in which, π_{ij} is the probability that the random surfer visits node p_j, assuming that it is currently visiting node p_i at the moment, and n is the number of nodes. In PageRank algorithm, these probabilities are the same in each row and calculated as follows:

$$\pi_{ij} = \frac{1}{|p_i|}, \quad \text{where } 1 \le j \le n$$

It means that in PageRank, the probability of moving from a page to each of its neighbours is the same. The matrix Π is the base for building the random surfer matrix in the PageRank algorithm. This matrix is also called *transition probability matrix*.

Our implementation of the random surfer is different from that employed in web page ranking and link recommendation systems such as (Agarwal, Chakrabarti, 2007; Backstrom and Leskovec, 2011). The typical random walk (Langville and Meyer, 2006) is an iterative process wherein, in each iteration the next node to be visited is selected randomly and uniformly. The random walker may traverse each node multiple times based on the number of incoming links of a node. It has been shown that in some instances convergence can take several iterations (Langville and Meyer, 2006). The proposed surfer is different since the probability of selecting a node as the next step is not the same for all available candidates. Some nodes have a higher chance than others to be visited by the random surfer. Moreover, the surfer takes a limited number of steps and the graph traversal concludes when a sufficient number of workers are recruited, or if it is not possible to go deeper. This significantly reduces the time complexity of the algorithm.

Assume that in a social network, member m_i serves as the requester and intends to publish a task. Let $\varphi_i = \{m_j | m_j \text{ is friend with } m_i\}$ be the set of m_i's friends. In order to find suitable participants, we initiate K random surfers where $K = |\varphi_i|$. Each random surfer, denoted by ω_j, starts from the friend m_j and walks through the graph to find and nominate suitable participants. Assume that the current state of a random surfer ω_j is m_{cur}. The random surfer first checks the suitability of m_{cur}. If the suitability score is greater than a predefined threshold, he will be invited to contribute. The surfer then continues its journey to find other nominees from the list of m_{cur}'s friends and φ_i is updated accordingly. The next step will be selected from φ_{cur} based on the suitability scores. In other words, the probability of selecting m^j_{cur} (the jth friend of m_{cur}) as the next step, denoted by $\pi_{\text{cur},j}$, is:

$$\pi_{\text{cur},j} = \frac{\sigma_j * \tau_{\text{cur},j}}{\sum_{k:k\in\varphi_{\text{cur}}} \sigma_k * \tau_{\text{cur},k}}$$

where σ_j is the m^j_{cur}'s suitability score and $\tau_{\text{cur},j}$ is the pairwise trust of m_{cur} upon his jth friend. It is evident that for each member m_i, the sum of probabilities of moving to his friends is equal to 1. In other words, $\sum_{j:1}^{K} \pi_{i,j} = 1$. Based on this, the stochastic matrix Π can be filled as $\Pi_{n\times n} = \{\pi_{i,j}\}$ in which, each element $\pi_{i,j}$ is the probability of selecting m_j as the next step for a random surfer that currently is in m_i. Π can be used by random surfers to determine the next steps.

The random surfer continues walking through the graph and inviting nominees to contribute. In order to control how far a random surfer can move from the requester, we define a parameter called the *propagation factor* and denote it by λ. The selection of an appropriate value for λ is challenging. A greater value of λ allows the random surfer to crawl deeper in the social graph, and thus increases the chance of finding more suitable workers. On the other hand, it may increase the risk of privacy leakage due to getting far from the requester's friendship network.

In the following, we present the practical implementation of the process discussed above, in the form of an algorithm. Algorithm 1 presents our proposed nomination algorithm. In this algorithm, W is a shared list which is accessible to all random surfers initiated for task θ_j, and includes the ID of all members who are invited by random surfers. Therefore, in each step, W contains the list of participants which have been nominated so far. This list is used as a shared memory among random surfers to prevent them from nominating a member twice. The algorithm first initialises an empty list W. It also extracts the list of all m_i's friends. Upon each friend m^j_i (jth friend of m_i), a separate random surfer ω_j is initiated with the current state set to be m^j_i (lines 1 to 8 in the algorithm).

The lines 9 to 28 are the steps that each random surfer takes independently. Each random surfer, ω, checks to see if its current state is suitable to contribute to the task, using the equations proposed in the Section 2.1.1. If suitable, he will be nominated. Then, the random surfer ω loads the row of Π corresponding to the current state of ω and then updates the transition probabilities. In order to do so, the pairwise trust between the current node and the nominee is investigated. If less than a specific threshold, m_j's suitability score will be set to zero. Otherwise, it will be updated with the value $\tau_{\text{cur},j}$ stated in Section 2.1.2. As can be seen, the probability of a particular member being visited by a surfer is in direct relationship with his suitability score.

2.2. *Selection module*

As a result of the nomination process, a finite set of eligible participants are nominated and invited to contribute. The selection module is responsible for selecting the participants from the set of nominees who have accepted the invitation. As mentioned in Section 1.2, our proposed selection module is aimed at overcoming the relevant issues in existing selection methods (i.e., open-call, auction, and pre-selection) by utilising the best of these recruitment strategies. In this module, we propose a time-aware and collusion-free recruitment method which is an extension to the pre-selection recruitment approach and aims at preventing the collusion.

2.2.1. *Eligibility assessment*

Whenever a nominee accepts the invitation, the selection module takes into account a set of time-aware parameters and decides whether to select the participant. The intuition behind considering these parameters is that when the task's remaining time is short, it is rational to select the nominee that has shown timely behaviour in his past contributions, to maximise the chance of receiving a contribution before the deadline. These parameters are (i) the selection score (i.e., the ratio of participants selected so far to the total number of required participants), (ii) the remaining time to the task deadline and (iii) the timeliness of the participant in previous tasks. The first two parameters are combined via a geometric mean function to form a time suitability score. Geometric mean is often used for comparing different items and finding a single "figure of merit" for these items, when each item has multiple properties. So, for the member m_i to be selected to attend in task θ_j, the time suitability will be as follows:

$$\text{Time Suitability}(m_i) = \sqrt{\text{Timeliness}(m_i) * \text{Remaining Time}(\theta_j)}$$

The time suitability is then combined with the selection score via a fuzzy inference engine. The result is an eligibility score for the nominee as follows.

$$\text{Eligibility Score}(m_i) = \text{Fuzzy}(\text{Time Suitability}(m_i), \text{Selection Score}(\theta_j))$$

If greater than a predefined threshold, the nominee will be considered to be eligible to participate.

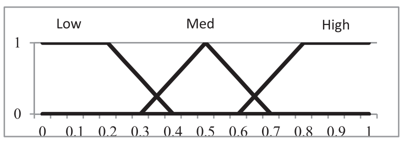
Fuzzy inference system. Our proposed framework employs fuzzy logic to calculate the Eligibility Score (ES) for each nominee. The use of fuzzy logic allows us to achieve a meaningful balance between the time suitability and the selection score. We cover all possible combinations of Time Suitability (TS) and Selection Score (SS) and address them by leveraging fuzzy logic in mimicking the human decision-making process. The inputs to the fuzzy inference system are the crisp values of TS and SS. The fuzzifier converts the crisp values of input parameters into a linguistic variable according to their membership functions. In other words, it determines the degree to which these inputs belong to each of the corresponding fuzzy sets. The fuzzy sets for TS, SS and ES are defined as:

$$T(TS) = T(SS) = \{\text{Low, Med, High}\},\ T(ES) = \{\text{VL, L, M, H, VH}\}.$$

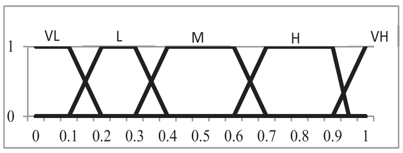
Fig. 1(a) represents the membership function of TS and SS and Fig. 1(b) depicts the ES membership function. We used trapezoidal shaped membership functions since they provide adequate representation of the expert knowledge, and at the same time, significantly simplify the process of computation.

The fuzzy inputs (TS and SS) are then converted to the fuzzy output (ES) by leveraging If-Then type fuzzy rules. The combination of the above mentioned fuzzy sets create 3*3 = 9 different states which have been addressed by 9 fuzzy rules as shown in Table 1(c). Fuzzy rules help in describing how we balance the various eligibility aspects. The rule base design has been done *manually*, based on the experience and beliefs on how the system should work (Yaghmaee et al., 2005). To define the output zone, we used *max-min* composition method. The result is ES which is a linguistic fuzzy value. Finally, in order to convert the ES fuzzy value to a crisp value in the range of [0, 1], we employ the Centre of Gravity (COG) (Leekwijck and Kerre, 1999) defuzzification method, which computes the centre of gravity of the area under ES membership function. COG is perhaps the most commonly used and popular defuzzification technique with the advantage of quick and highly accurate computations.

Once the crisp value for the eligibility score is computed, it is compared to a predefined threshold (has been set to 0.5 in



(a) Membership function for TS and SS



(b) Membership function for ES

Rule no.	if TS	and SS	then ES
1	Low	Low	M
2	Low	Med	L
3	Low	High	VL
4	Med	Low	H
5	Med	Med	M
6	Med	High	L
7	High	Low	VH
8	High	Med	H
9	High	High	M

(c) Fuzzy rule base

Fig. 1 – Rule base and Membership functions of input and output linguistic variables.

Algorithm 1 Nomination Algorithm

Input: Π as the transition probability matrix, m_i as the requester, NoP is the required number of participants, and θ_j as the advertised task

Output: W as the list of nominees.

```
1:  $\varphi_i$ = list of $m_i$'s friends
2:  Initialise $W$ as an empty list.
3:  for all $f \in \varphi_i$ do
4:      Initiate a random surfer $\omega_j$ from $f$
5:      //current state of $\omega_j$ is $f$
6:  end for
7:  $\Omega$ = set of all initiated random surfers
8:  for all $\omega \in \Omega$ do
9:      L=$\lambda$
10:     while true do
11:         // $m_{cur}$ denotes the current state of the random surfer $\omega$
12:         if $m_{cur}$ is suitable for $\theta_j$ then
13:             if $|W| <= 2 * NoP$ then
14:                 Nominate $m_{cur}$
15:                 Add $m_{cur}$ to $W$
16:             end if
17:         else
18:             Stop random surfer $\omega$
19:             Exit
20:         end if
21:         Load $\pi = \Pi_{cur}$ //the row of $\Pi$ corresponding to the current node
22:         Update $\pi$ // see the algorithm description for details
23:         if $\pi$ is empty then
24:             // there are no choices for next step
25:             Stop random surfer $\omega$
26:             Exit
27:         end if
28:         Select a member of $\pi$ as $m_{cur}$ // see the algorithm description for details
29:         L = L+1
30:         if $L \geq \lambda$ then
31:             Stop random surfer $\omega$
32:         end if
33:     end while
34: end for
35: return $W$
```

the implementation). If greater than the threshold, the nominee is considered as eligible.

2.2.2. *Collusion prevention*

Once the member m_i is considered to be eligible for being selected, a final check is done to ensure that the selection of m_i will not result in potential collusion. In particular, we aim to identify whether the addition of m_i to the set of previously selected participants will result in the formation of a group of colluders.

Collaborative attacks which are also called collusion attacks are those in which, a group of people collaborate on changing the results of a task (Mukherjee et al.). For example, they may collaborate to be selected as the final participants and then, produce poor quality contributions that severely impact the goal of the task. Most existing collusion detection techniques rely on behavioural indicators to identify colluding groups (Allahbakhsh et al., 2013; Mukherjee et al.; Lim et al., 2010). These indicators reflect suspicious behaviour from a group of members which indicates the possibility of collusion. For example, the collaboration of a group of participants may be considered as collusion if the following suspicious behaviours are observed: (i) R members of the group have collaborated (ii) these members attend in the same k tasks; (iii) they submitted their contributions within a small time window. All these factors occurring together strongly suggest suspicious activities.

In our selection module, for each new participant to be selected, we consider a set of indicators that suggest the likelihood of the formation of a colluding group among the selected participants. Note that these indicators reflect the likelihood of collusion only when they all occur together. The first indicator is the Group Size (GS) which is the number of colluders who collaborate in similar tasks. The larger the group, the more damaging it is. The second indicator is the Group Support Count (GSC) which is the number of tasks in which the group members have collaborated in the past. Groups with high support counts are more likely to be colluding as the probability of a group of random people to have attended the same tasks together is rather small. The third indicator is the Group Time Window (GTW) which indicates the time window of the group contributing to a task. A group of participants contributing to a task within a short burst of time is more prone to be colluding. These indicators happening together indicate the collusion possibility. So, for each eligible participant m_i to be selected, these indicators are investigated. If all greater than certain related thresholds, it implies that the selection of m_i may lead to potential collusion, and hence, the participant will not be selected.

In order to compute the possibility of collusion, the first step is to identify all existing subgroups in the group of selected participants (including m_i), who have collaborated on multiple tasks in the past. To do so, we use the well-known technique called the Frequent Itemset Mining (Grahne and Zhu, 2005) which has performed well for collusion detection in previous literature (Mukherjee et al.; Allahbakhsh et al., 2013). In our context, a set of items are the set of all selected participants for the current task. The set of transactions are the set of all tasks that m_i has been involved in the past. By mining frequent itemsets, we find groups of participants who have contributed to multiple tasks together. We consider a group as a tuple in the form of $(\gamma_i^\theta(P), \gamma_i^\theta(T), \gamma_i^\theta(C))$. $\gamma_i^\theta(P)$ is the set of group members' IDs, $\gamma_i^\theta(T)$ is the list of all tasks these members has collaborated in, and $\gamma_i^\theta(C)$ is the set of all contributions made by members to these tasks. Each contribution has a timestamp indicating its submission time. The difference between the latest and earliest timestamp of contributions submitted by the group members indicates the group time window, denoted by $\gamma_i^\theta(\varepsilon)$. Based on the FIM output, the indicator values can be quantified as follows. Group Size (GS) $= |\gamma_i^\theta(P)|$, Group Support Count (GSC) $= |\gamma_i^\theta(T)|$, and Group Time Window (GTW) $= \gamma_i^\theta(\varepsilon)$.

We consider a group of collaborators collusive, if all the following conditions are met:

1. If the Group Size (GS) is greater than a predefined threshold th_1.
2. If the Group Support Count (GSC) is greater than a predefined threshold th_2.
3. If Group Time Window (GTW) is smaller than th_3.

The member m_i will be selected to contribute to the task θ if no group with the above mentioned conditions is created as a result of this selection.

3. Experimentation and evaluation

In this section, we conduct a simulation-based evaluation to analyse the behaviour of our proposed framework. First, we explain experimentation setup, the metrics we use for performance evaluation and the datasets we used in experiments in Section 3.1. Then, we compare our proposed framework with other methods in Section 3.2. Then, we analyse the behaviour of our framework in Section 3.3 in order to find an optimum configuration. Finally, in Section 3.4, we investigate the efficiency of our proposed collusion prevention method.

3.1. *Experimentation setup*

To undertake the preliminary evaluations outlined herein, we chose to conduct simulations, since real experiments in social participatory networks are difficult to organise. Simulations afford a controlled environment where we can carefully vary certain parameters and observe the impact on the system performance. Our simulations have been conducted on a PC running Windows 7.0 professional and having 4 GB of RAM. We used Matlab R2012 for developing the simulator.

3.1.1. *Dataset*

The dataset that we use for our experiment is the real web of trust of Advogato.org (Levien and Aiken, 1998). Advogato.org is a web-based community of open source software developers in which, site members rate each other in terms of their trustworthiness. The result of these ratings among members is a rich web of trust, which comprises of 14,019 users and 47,347 trust ratings. The Advogato web of trust may be viewed as a directed weighted graph, with users as the vertices and trust ratings as the directed weighted edges of the graph. So, it is in perfect match with our assumptions related to participants and their trust relations in social participatory sensing. We also pre-processed the dataset in order to remove the isolated nodes that have no connections. 174 nodes were identified as isolated and were removed.

To use this graph as a social participatory sensing system, members should have attributes reflecting their social behavioural factors such as the reputation score, number of expertise, the probability of participating in a task, etc. We have computed a reputation score for each member by calculating the average of all pairwise trust scores the member received from his friends. The reputation score is a number in the range of [0, 1]. In order to simulate a real-world scenario and assign real values to other attributes, we enriched the Advogato dataset using the Stackoverflow dataset[6] created by the Stack Exchange, Inc. In other words, we extract various statistical parameters from Stackoverflow and use them as a guide to assign values to nodes' main attributes in Advogato graph. We use the Programmer subset of the Stackoverflow which contains 30,060 posts (both questions and answers) created by 12,081 users. These users have earned 24,863 badges based on their activities in the community. The badges reflect the expertise of the participants.

Using the standard statistical estimation tools provided by Matlab® software package, we have determined that in Stackoverflow dataset, number of badges per participant follows a normal distribution $X = \mathcal{N}(5.57, 6.52)$ (where X is a random variable). Similarly, number of tasks per participant (i.e., number of tasks each participant has attended in) and

[6] https://archive.org/details/stackexchange.

number of participants per task follow the normal distributions $Y = \mathcal{N}(6.39, 18.4)$ and $Z = \mathcal{N}(4.25, 6.75)$, respectively.

We used the above knowledge to initialise attributes for participants. Specifically for ‘number of expertise’, we utilise the random variable X and assign each participant a value k representing their number of expertise. We then uniformly select k numbers from the range of (Murugesan, 2007; Allahbakhsh et al., 2013) as expertise values. The intuition behind this selection is that based on literature, approximately twenty different types of tasks can be identified in a typical crowdsourcing system[7] (Doan et al., 2011). The participant's expertise reflects how good he is in accomplishing tasks of various types. We also utilised the Y variable to assign each participant a value representing the ‘number of attended tasks’. Then we use the assigned number to compute the participant's probability of contribution in a task. This probability is computed based on the number of tasks in which a worker has participated and the total number of tasks in the system. Finally, we use the random variable Z to assign ‘number of participants’ to each generated task.

In order to evaluate the performance of our proposed collusion prevention method, we utilised the Wikipedia voting dataset. In Wikipedia,[8] the voting process is used to elect administrators.[9] Every registered user can nominate himself or another user as an administrator in Wikipedia and initiate an election. The other users participate in the election and cast their votes on the eligibility of nominee. If the majority of users recognise a user as eligible, then this user becomes a Wikipedia administrator. In order to incorporate this dataset in the context of our framework, we employ the following mapping. The requester is the nominee, the worker is the voter, the task is evaluating the eligibility of the nominee as an administrator in Wikipedia and the contribution is the worker's vote. We use the log of Wikipedia Adminship Election[10] which is collected by Leskovec et al. for behaviour prediction in online social networks (Leskovec et al., 2007), referred to as WIKILog. WIKILog contains about 2800 elections (tasks) with around 100,000 total votes and about 7000 users participating in the elections either as a voter or a nominee. We use the WIKILog to demonstrate the efficacy of our proposed framework to detect collusion.

3.1.2. *Evaluation method and metrics*

In order to evaluate the performance of our proposed framework, we run the experiment for a set of rounds. A simple experimentation round contains the following steps: In the first step, we choose a requester out of the members of Advogato community. This selection is performed uniformly, meaning that all members have the same chance to be chosen as the requester. Then, a task is generated to be advertised to the community. Each task contains a set of attributes, mainly, a minimum accepted reputation score, a set of at most 5 required expertise attributes, and the maximum number of required participants. Once the requester is chosen and the task is generated, the nomination algorithm (Algorithm 1) is executed in order to find and invite suitable nominees. Then the selection module, explained in Section 2.2, chooses a subset of nominees as selected participants to contribute to the task. We assumed that at least 50% of nominees apply to the task for contribution.

In order to evaluate the effectiveness of framework modules, we define four evaluation metrics. The first metric is the *number of nominees*. The ability to identify more suitable nominees is a desirable property of the nomination module. The second evaluation metric is the *overall suitability score of the nominees*, which is the average of all nominees' suitability scores. A larger value for this metric suggests that the nomination module is able to recruit well-suited participants. We have two similar metrics to evaluate the performance of the selection module: the *number of selected participants* and the *overall suitability score of selected participants*. All results shown in charts are the average of outcome of running the experiment for 1000 independent rounds.

3.2. *Performance comparison*

In this section, we compare the performance of our framework with two well-known recruitment methods: (i) *Open-call* which is used in most existing crowdsourcing platforms such as Amazon Mechanical Turk,[11] CrowdFlower,[12] etc. Recall that in this scheme, the requester broadcasts the task to all members in the community and everyone is able to contribute to the task. (ii) *Friend-based* which is widely used in social networks and related work such as (Emek et al., 2011; Kleinberg and Raghavan, 2005), wherein, the requester advertises the task to his friends.

It should be noted that neither of these methods consider the privacy preservation in their recruitment. So, in order to have a fair comparison, we consider each of the compared methods with two separate configurations: privacy-aware and non privacy-aware. In privacy-aware configuration, the weights of reputation score, expertise, privacy score and requester's preference in the computation of the suitability score are 0.5, 0.3, 0.1 and 0.1, respectively (refer to the Section 2.1.1). In non privacy-aware recruitment, the weight of privacy score is zero and reputation, expertise and preference are taken into account with weights of 0.55, 0.35, and 0.1 respectively. Another important point is that the simulation results illustrated in the following figures have been scaled with the number of participants in order to have reasonable comparisons. For example, the number of selected participants for each of the aforementioned methods has been scaled with the corresponding maximum number of participants. In order to evaluate the performance of methods in real situations, we run the simulation in three different situations: (i) when the requester has few friends, (ii) when the requester has large number of friends, and (iii) when we select the requester randomly, regardless of his number of friends. Note that the concepts such as ‘few’ or ‘large’ are relative and depend on the characteristics of the underlying social network. In order to

[7] http://gigaom.com/2010/11/19/18-tasks-you-can-crowdsource/.
[8] http://www.wikipedia.org/.
[9] http://en.wikipedia.org/wiki/Wikipedia:Requests_for_adminship.
[10] http://snap.stanford.edu/data/wiki-Elec.html.
[11] http://mturk.com.
[12] http://crowdflower.com.

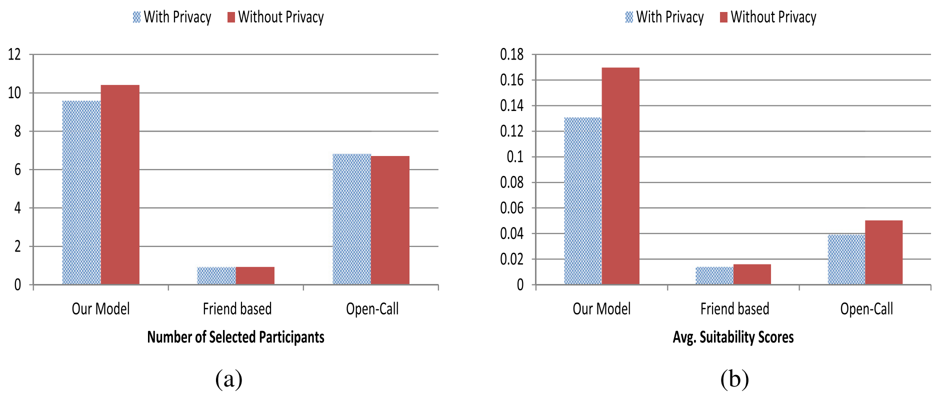


Fig. 2 – Performance of three methods in the case of requesters with few friends.

consider these situations, we first arrange all members (i.e., Advogato members) in ascending order according to their number of friends (outgoing links). For the first situation, the requester will be selected from the first one third of the members, and for the second situation, the requester will be selected from the last one third. The last situation will be the case when the requester is selected randomly and uniformly from the unordered list. In Advogato, the range of number of friends for the first group is between 3 and 1000, and for the second group is between 3000 and 4000. To come up with dependable results, we run the simulation for 1000 rounds.

Fig. 2(a) and (b) depict the results of comparing three methods for the case in which, the requester has few friends. As it is evident from the charts, our model outperforms other two methods in terms of the number of selected participants. This is an expected result since in our method, we consider the suitability of participants while selecting them to attend, whereas some of those who are nominated by other methods may not be suitable. Also, when it comes to overall suitability score of the selected participants, the best performance belongs to our framework. This is because our method considers the suitability scores for in selection module and tries to assign higher selection probability to participants with higher suitability scores. This better performance is of great importance since it demonstrates a valuable achievement for the case of having sparse friendship network, which, as mentioned in Section 1.2, is currently an issue in existing online social networks. The relative order of these 3 methods is consistent in both privacy-aware and non privacy-aware scenarios.

Fig. 3(a) and (b) illustrate the performance of three methods for the case in which, the requester has large number of friends. In this case, as it is expected, the performance of the friend-based method improves since the number of friends (potential participants) has increased for both (privacy-aware and non-privacy-aware) scenarios. The best performance still belongs to our framework. This is due to the large number of friends in the requester's friendship graphs, which in turn, increases the number of random surfers and consequently, the number of selected participants. Finally, Fig. 4(a) and (b) show the results of our experiments when we selected a requester from the community, regardless of his number of friends. In this case, as it is expected, due to the scarcity of the social network, the overall performance of the open-call is better than friend-based method. The best performance still belongs to our proposed method.

As all above figures show, the number of nominees and selected participants decreases when privacy considerations are taken into account. This decrement is expected and is because such considerations will result in tighter restrictions in selection module. The important point is that relative ordering of the methods in terms of performance remains unchanged in both the privacy-aware and non-privacy-aware scenarios.

3.3. *Sensitivity analysis*

In this section, we run a series of experiments to reach to an optimal setting for our proposed recruitment framework. In particular, we first obtain the optimal value for λ (propagation factor), and then evaluate the performance of our framework in the presence/absence of privacy considerations and participant selection process.

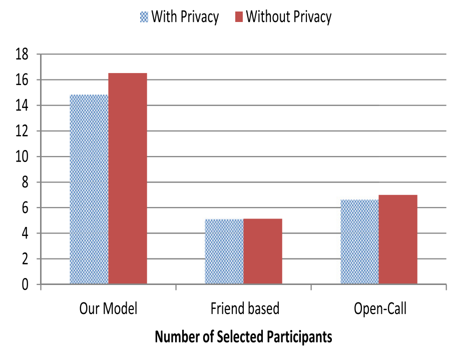
3.3.1. *Optimum value of λ*

One of the important parameters which impacts the performance of the random surfer is the propagation factor, denoted by λ which denotes how deep the random surfer can explore the graph to find suitable participants. In order to assess the performance of our framework, we need to find an optimum value for λ. Note that λ is a system-dependant parameter and its optimum value totally depends on the characteristics and the size of social graph. We conducted an experiment to test the framework on Advogato graph with different values of λ in the range of 1–150. For each λ value, we generated 500 tasks and then investigated the outcomes. Based on various runs of this experiment, the highest value of overall suitability score for selected participants is obtained when λ is equal to 100. So, we select this value for λ as the optimum value for future experiments.

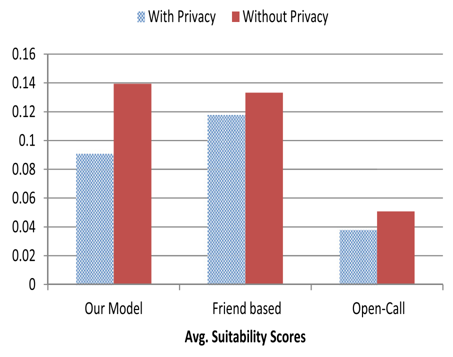
3.3.2. *Performance analysis of framework components*

In addition to the value of λ, we investigate the impact of two other aspects on the performance of our proposed framework. The aim of these experiments is to obtain the best configuration for our proposed framework.

At first, we try to investigate the effect of privacy score in the evaluation of suitability score. As mentioned before in Section



(a)



(b)

Fig. 3 – Performance of three methods in the case of requesters with large number of friends.

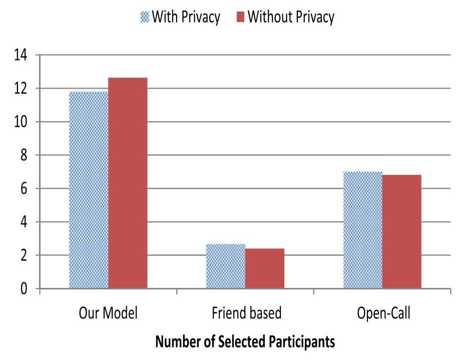
2.1.1, the probability of a selecting a non-friend participant for further tasks of a particular requester has an inverse relation to the number of the tasks he has been involved for that requester, due to the reduction in his privacy score. In other words, taking the privacy into consideration, while valuable in terms of members' security, will inherently decrease the number of potential participants. In the following experiments, we aim at investigating how the privacy score consideration will affect the framework performance in terms on number of nominees and selected participants.

Next, we aim at observing the performance of our framework with and without the selection module. We expect that including the selection module will increase the overall suitability score, but at the same time, will decrease the number of final participants, since it tightens the criteria of participant selection.

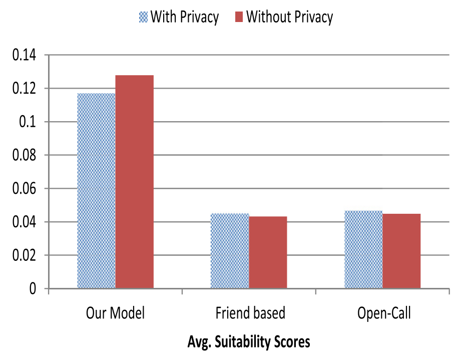
In order to evaluate the effect of these two components, we conducted an experiment in which, the performance of our framework is evaluated with the following four scenarios:

1. In the first scenario, we neither take privacy nor selection module into account. In other words, the suitability score of nominees is only calculated based on their reputation, expertise and the requester's preferred and blocked list. Also, we deactivate the selection module. In our illustrations in Fig. 5(a) and (b), we represent this scenario by '*NONE*'.
2. In the second scenario, the selection module is active and working. The privacy does not affect the suitability score. We denote this scenario by '*S*'.
3. In the third scenario, the privacy score is considered in the evaluation of suitability scores. The selection module, however, is not included, meaning that when a nominee applies to do a task, no restriction will be applied and he will be directly accepted if there are still vacant places. This scenario is denoted by '*P*'.
4. The forth scenario, is our proposed framework where both privacy and selection aspects are taken into account. We denote this scenario by '*PS*'.

The evaluation results are depicted in Fig. 5(a) and (b). As shown in Fig. 5(a), the overall number of nominees and selected participants is the highest in the first scenario (when we have neither selection module nor privacy considerations). This is because both the privacy consideration and the involvement of selection module pose limitations in the number of nominees and selected participants. However, as Fig. 5(b) reveals, the overall suitability score in this scenario is too low, since there is no suitability check in the selection process. So, it cannot be deemed as a good configuration. The same argument can be applied to the third scenario as well. In this scenario, the number of selected participants is greater than those methods which include the selection module, since there is no limitation for participant selection. However,

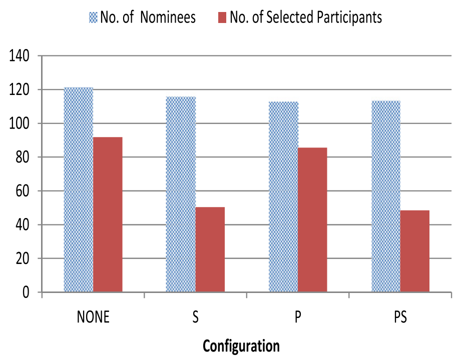


(a)

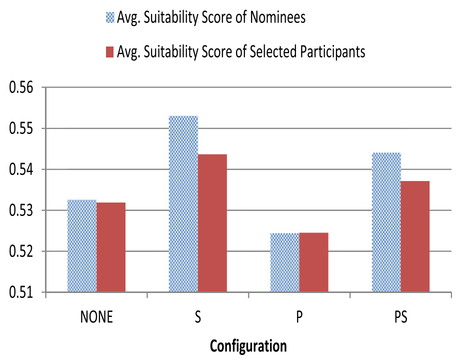


(b)

Fig. 4 – Performance of all methods regardless of requesters' number of friends.



(a)



(b)

Fig. 5 – Evaluation of our framework with different scenarios.

the average suitability score in this scenario is the least, compared to other scenarios, since it does not consider the suitability score as a dominant factor. So the optimum configuration is to be selected from the second and forth scenarios (S and PS scenario). In both S and PS scenarios, the selection process is applied; but privacy is considered only in PS. The overall number of nominees and selected participants in both settings are approximately the same, but the overall suitability score in S scenario is slightly (about 0.009) higher than the suitability score in PS. Therefore, we conclude that PS configuration is the best for the privacy-aware systems and S configuration is appropriate for the rest.

3.4. *Collusion prevention analysis*

As mentioned in Section 3.1, we use Wikipedia Adminship Election dataset to investigate the performance of our proposed collusion prevention method. The dataset contains the information related to 2794 tasks. The average number of participants in these tasks equals to 40. In order to obtain reliable results, we considered the tasks with number of participants greater than the average as the sample data, and randomly select 100 tasks from these. We then tested our proposed method to identify any potential colluding group among the participants. As mentioned in Section 2.2.2, we considered three indicators for detecting potential collusion. More precisely, we assume that a group of participants is considered as a colluding group if it has at least th_1 members who all have collaborated in at least th_2 tasks in the past, and they have all submitted their contributions to these tasks in time window not larger than th_3.

In order to find the optimum value for th_2, we set an experiment in which, the target size (i.e., number of the tasks that the group members have collaborated in them in the past) is changed. For each target size, we measure the number of groups identified, together with their size. As can be seen in Fig. 6, the maximum size of identified groups decreases by increasing the target size. This is rational since the probability of finding groups whose members have collaborated in greater number of tasks is smaller. We believe that the best setting is the one which results in the identification of largest groups to make a considerable impact. As derived from the figure, this situation is related to the case where the target size is 6. So, we set th_2 to be equal to 6. For the sake of simplicity, we assume that th_1 equals to th_2. th_3 has also been set to the average of all time windows for all the tasks. So, our method aims at identifying the set of candidate groups who have at least 6 members, have all collaborated in at least 6 tasks in the past and submitted their contributions within a specific time window.

In order to investigate the performance of our proposed collusion prevention method, we first utilised the FIM technique to find the candidate groups among the participants. The outcome was the discovery of 18 candidate groups with at least 10 members. We then employed our collusion prevention method and identified 9 of these 18 groups as collusive. To evaluate the efficiency and accuracy of our method, we went through a set of statistical calculations. At first, we measured the ratio of the tasks targeted with the colluding groups. The result shows that 14% of the tasks were affected by these 9 colluding groups. This means that our collusion prevention method is able to prevent 14% of the tasks from being targeted by the colluders. We then calculated the success ratio of the tasks targeted by the colluding groups as well as all 100 tasks. By success ratio, we mean the ratio of the tasks that have resulted in a decision, to the total number of tasks (note that in the Wikipedia adminship election dataset, a task (an election) is successful if it results in the selection of the user as an administrator). We observed that overall success ratio of the tasks in our dataset is 71%. This ratio is 83% for the groups identified by our collusion detection method. This means that there is a high probability that the groups identified by our method are colluding groups, since their collaboration has resulted in a considerably high success ratio. This is a significant indication that the identified groups are much likely to be collusive.

To be brief, the results show that our proposed collusion prevention method is successful in preventing the formation of colluding groups among the selected participants with high accuracy. This is due to the correct selection of indicators as well as accurate settings of the thresholds.

4. Related work

Social participatory sensing can be regarded as a subset of collective intelligence systems, which are defined broadly as

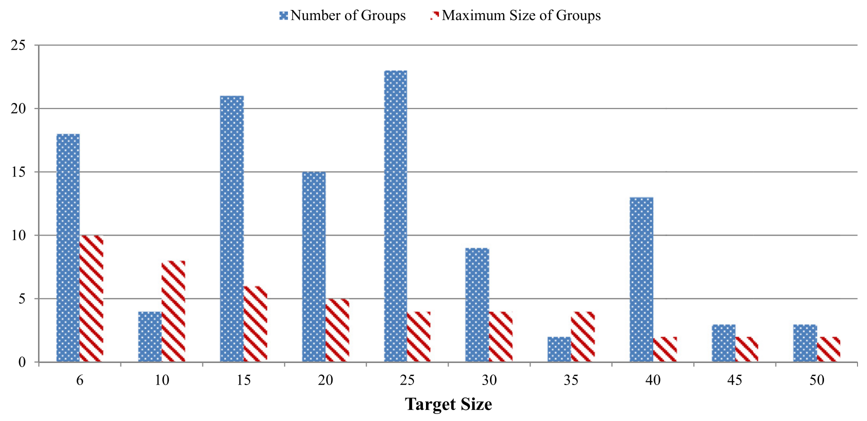


Fig. 6 – Evolution of number of groups and their maximum size according to the target size.

groups of individuals doing things collectively that seem intelligent (Malone et al., 2009). Due to the openness of such systems, the recruitment of sufficient well-suited participants has always been a great concern (Kanhere, 2013). In the following, we will have a short review on the related works on this issue and will discuss the state-of-the-art.

One of the important parameters in obtaining high quality contributions is the effectiveness of the methods utilised for participant selection. It is evident that the volume and the diversity of participants with different perspectives and knowledge can lead to accurate trustworthy contributions.

In most online recruitment systems such as CrowdFlower, Wikipedia, etc., the recruitment process includes a nomination step in which, nominees are selected amongst the crowd based on a set of criteria. Sometimes, there is no explicit nomination. In such systems, the requester advertises the task to the crowd (similar to writing on Facebook wall) and everyone inside the system is able to contribute. Example is Amazon Mechanical Turk (Murk), which by default, outsources a task to everyone within the system in the form of an open call.

This open-call method, however, may lead to poor fidelity contributions, since it enables anyone, even those poorly equipped to fulfil the task, to contribute. Sometimes the requester may wish to specify that workers possess certain attributes in order to complete the task. For example, participant selection in MTurk can be restricted to residents of a specific country, or to workers who have completed more than a certain number of tasks with a specified rate of accuracy. Thus, smaller bespoke crowds can be assembled out of the workforce to complete highly specialised tasks (Chandler et al., 2013).

On the other hand, sometimes, the task assignment is done on the basis of a publish/subscribe service, i.e., the participant (subscriber) shares his interests and preferences about a topic by subscribing to the server and a requester (publisher) posts and forwards messages to the interested users only (Demirbas et al., 2010; Chatzimilioudis et al., 2012). The main challenge in the above nomination methods is the lack of sufficient qualified participants to attend in the tasks that need specific knowledge or expertise (Chandler et al., 2013; Doan et al., 2011).

In social participatory sensing, where the social relationships are utilised to find and nominate the eligible participants, *random walk* (Spitzer, 2001) can be used for participant nomination. The concept of random walk originates from graph theory (Lovász, 1993), and has a wide range of applications. One of the important examples is the link prediction algorithms. Given a large network, say Facebook, at time *t*, for each user, link prediction algorithm is aimed at predicting what new edges (friendships) that user will create between *t* and some future time t_1 (Backstrom and Leskovec, 2011). Similarly, link recommendation algorithms aim to suggest to each user a list of people that the user is likely to create new connections to (Konstas et al., 2009). Random walk has also been used for community detection in online communities. Latapy and Pons, 2004 leverage the idea that short length random walks on a graph tend to get “trapped”into densely connected parts corresponding to communities. Sadilek et al., 2013 leverage the idea of random walk for crowdsourcing and routing tasks that require people to collaborate and synchronise both in time and physical space. Graph sampling (Kurant et al., 2011) and node ranking (Agarwal et al., 2007) are examples of other applications for random walk. In our framework, we leverage the concept of random walk for nominating the eligible participants. In particular, we propose a customised random surfer which is responsible for crawling the social graph and identifying the suitable candidates as nominees. Our proposed random surfer, however, is different from the typical random walk, as it does not select the next step purely random, but based on a probability matrix. Moreover, despite the typical random walk which continues until convergence happens, our proposed random surfer takes a limited number of steps to find well suited participants amongst the friendship network. This will reduce the complexity of our proposed nomination solution.

As mentioned earlier, if the open-call strategy is used for nomination, there is no need for selection as everyone can contribute. However, for some crowdsourcing systems such as oDesk,[13] there exist restrictions in the number of participants. In such cases, a limited number of participants should be selected from the set of nominees who have been identified by the nomination method. This restriction may be due to the nature of the task itself (such as time-critical tasks) (Nath et al., 2012), limitation in the resources needed for incentivising the participants and/or evaluating the contributions (cost-critical tasks) (Nath et al., 2012). In order to select from the nominees, the pre-selection methods select a fixed

[13] http://odesk.com.

number of participants to compete to contribute (Vukovic and Bartolini, 2010). Our proposed selection module is different from the above mentioned methods as it restricts the number of nominees, and at the same time, gives priority to well-suited nominees to be selected as final participants.

Collusion detection has been widely studied in P2P systems (Ciccarelli and Lo Cigno, 2011; Lian et al., 2007). A comprehensive survey on collusion detection in P2P systems can be found in (Ciccarelli and Lo Cigno, 2011). Reputation management systems are also targeted by collusion. Colluders in reputation management systems try to manipulate reputation scores by collusion. Many efforts are put into detecting collusion using majority rules, weight of the worker and temporal analysis of the behaviour of the users (Sun and Liu, 2012) but none of these methods is strong enough to detect all sorts of collusion (Sun and Liu, 2012). Mukherjee et al. have proposed a model for spotting fake review groups in online rating systems. The model analyses textual feedback cast on products in Amazon's online market to find collusion groups. They use eight indicators to identify colluders and propose an algorithm for ranking collusion groups based on their degree of spamicity. However, their proposed method is still vulnerable to some attacks. For example, if the number of attackers is much higher than honest raters on a product the model cannot identify this as a potential case of collusion. In the domain of participatory sensing, Huang et al., 2011 aim at detecting the collusion by leveraging a reputation management system and outlier detection algorithms. In (Dua et al., 2009), a trusted platform module (TPM) is provided with each sensor device to attest the integrity of sensor readings. This local integrity checking makes the system resistant to collusion. To the best of our knowledge, the collusion prevention has not been discussed in social participatory sensing, and the methods proposed for participatory sensing are not applicable to this domain.

5. Conclusion

In this paper, we proposed a participant selection framework for social participatory sensing. Our system leverages a customised random surfer to crawl the multi-hop friendship relations and identify well-suited nominees. The system then selects the final participants among the nominees. The selection is done in a way that it prevents the formation of a group of colluders within the set of selected participants. Using real world datasets, simulations demonstrated that our scheme increases the number of participants who are reputable and well-suited to contribute. It also performs well in efficient and accurate detection of colluding groups.

REFERENCES

Agarwal A, Chakrabarti S. Learning random walks to rank nodes in graphs. In: 24th international conference on Machine learning; 2007. p. 9–16.

Alkouz A, De Luca EW, Albayrak S. Latent semantic social graph model for expert discovery in Facebook. In: IICS, USA; 2011. p. 128–38.

Allahbakhsh M, Ignjatovic A, Benatallah B, Beheshti SMR, Foo N, Bertino E. Representation and querying of unfair evaluations in social rating systems. Comput Secur 2014;41:68–88.

Allahbakhsh M, Ignjatovic A, Benatallah B, Beheshti SMR, Bertino E, Foo N. Collusion detection in online rating systems. In: APWeb; 2013. p. 196–207.

Amintoosi H, Kanhere SS. A trust framework for social participatory sensing systems. In: MobiQuitous; 2013. p. 237–49.

Amintoosi H, Kanhere SS. A trust-based recruitment framework for multi-hop social participatory sensing. In: DCOSS; 2013. p. 266–73.

Amintoosi H, Kanhere S. A reputation framework for social participatory sensing systems. MONET 2014;19(1):88–100.

Amintoosi H, Kanhere S. Privacy-aware trust-based recruitment in social participatory sensing. In: MobiQuitous; 2014. p. 262–75 [in press].

Backstrom L, Leskovec J. Supervised random walks: predicting and recommending links in social networks. In: WSDM; 2011. p. 635–44.

Burke JA, Estrin D, Hansen M, Parker A, Ramanathan N, Reddy S, Srivastava MB. Participatory sensing. In: WSW workshop, SenSys; 2006.

Ciccarelli G, Lo Cigno R. Collusion in peer-to-peer systems. Comput Netw 2011;55(15):3517–32.

Chandler J, Paolacci G, Mueller P. Risks and rewards of crowdsourcing marketplaces. In: Handbook of human computation; 2013. p. 377–92.

Chatzimilioudis G, Konstantinidis A, Laoudias C, Zeinalipour-Yazti D. Crowdsourcing with smartphones. IEEE Internet Comput 2012;16(5):36–44.

Demirbas M, Bayir MA, Akcora CG, Yilmaz YS, Ferhatosmanoglu H. Crowd-sourced sensing and collaboration using twitter. In: WoWMoM; 2010. p. 1–9.

Doan A, Ramakrishnan R, Halevy AY. Crowdsourcing systems on the world-wide web. Commun ACM 2011;54(4):86–96.

Dua A, Bulusu N, Feng WC, Hu W. Towards trustworthy participatory sensing. In: Usenix Workshop on Hot Topics in Security (HotSec); 2009.

Ehrlich K, Lin CY, Griffiths-Fisher V. Searching for experts in the enterprise: combining text and social network analysis. In: ACM GROUP, USA; 2007. p. 117–26.

Emek Y, Karidi R, Tennenholtz M, Zohar A. Mechanisms for multi-level marketing. In: 12th ACM conference on Electronic commerce; 2011. p. 209–18.

Grahne G, Zhu J. Fast algorithms for frequent itemset mining using fp-trees. IEEE TKDE 2005;17(10):1347–62.

Huang KL, Kanhere SS, Hu W. A privacy-preserving reputation system for participatory sensing. In: IEEE LCN; 2012. p. 10–8.

Huang KL, Kanhere SS, Hu W. On the need for a reputation system in mobile phone based sensing. Ad Hoc Networks 2014;12:130–49.

Kanhere SS. Participatory sensing: crowdsourcing data from mobile smartphones in urban spaces. In: ICDCIT; 2013. p. 19–26.

Kleinberg J, Raghavan P. Query incentive networks. In: 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS); 2005. p. 132–41.

Konstas I, Stathopoulos V, Jose JM. On social networks and collaborative recommendation. In: ACM SIGIR; 2009. p. 195–202.

Krontiris I, Freiling FC. Urban sensing through social networks: the tension between participation and privacy. In: ITWDC, Italy; 2010.

Kurant M, Gjoka M, Butts CT, Markopoulou A. Walking on a graph with a magnifying glass: stratified sampling via weighted random walks. In: ACM SIGMETRICS; 2011. p. 281–92.

Langville AN, Meyer CD. Google's PageRank and beyond: the science of search engine rankings. Princeton University Press; 2006.

Latapy M, Pons P. Computing communities in large networks using random walks. 2004. arXiv preprint cond-mat/0412368.

Leekwijck WV, Kerre EE. Defuzzification: criteria and classification. Fuzzy Set Syst 1999;108(2):159–78.

Leskovec J, Adamic L, Huberman B. The dynamics of viral marketing. TWEB 2007;1(1).

Levien R, Aiken A. Attack-resistant trust metrics for public key certification. In: 7th USENIX Security Symposium; 1998. p. 229–42.

Lian Q, Zhang Z, Yang M, Zhao BY, Dai Y, Li X. An empirical study of collusion behavior in the maze p2p file-sharing system. In: ICDCS; 2007.

Lim EP, Nguyen VA, Jindal N, Liu B, Lauw HW. Detecting product review spammers using rating behaviors. In: CIKM; 2010.

Lovász L. Random walks on graphs: a survey, vol. 2, no. 1; 1993. p. 1–46.

Malone T, Laubacher R, Dellarocas C. Harnessing crowds: mapping the genome of collective intelligence. 2009. MIT Sloan Research Paper.

Mukherjee A, Liu B, Glance N. Spotting fake reviewer groups in consumer reviews. In: Proceedings of the 21st international conference on World Wide Web.

Murugesan S. Understanding web 2.0. IT Prof 2007;9(4):34–41.

Nath S, Dayama P, Garg D, Narahari Y, Zou J. Mechanism design for time critical and cost critical task execution via crowdsourcing. In: Internet and network economics; 2012. p. 212–26.

Page L, Brin S, Motwani R, Winograd T. The pagerank citation ranking: bringing order to the web. In: Technical report, Stanford Digital Library Technologies Project; 1999.

Reddy S, Estrin D, Srivastava M. Recruitment framework for participatory sensing data collections. Pervasive Comput 2010:138–55.

Satzger B, Psaier H, Schall D, Dustdar S. Auction-based crowdsourcing supporting skill management. Inform Syst 2013;38(4):547–60.

Sadilek A, Krumm J, Horvitz E. Crowdphysics: planned and opportunistic crowdsourcing for physical tasks, vol. 21, no. 10; 2013. 424, pp. 125–620.

Spitzer F. Principles of random walk. Springer; 2001.

Sun Y, Liu Y. Security of online reputation systems: the evolution of attacks and defenses. Signal Process Mag 2012;29(2):87–97.

Surowiecki J. The wisdom of crowds. Random House Digital, Inc; 2005.

Uger J, Karrer B, Backstrom L, Marlow C. The anatomy of the Facebook social graph. 2011. arXiv preprint arXiv:1111.4503.

Vukovic M, Bartolini C. Towards a research agenda for enterprise crowdsourcing. In: Leveraging applications of formal methods, verification, and validation; 2010. p. 425–34.

Yaghmaee MH, Menhaj MB, Amintoosi H. Design and performance evaluation of a fuzzy based traffic conditioner for differentiated services. Comput Netw 2005;47(6):847–69. Elsevier.

Yang SH, Long B, Smola A, Sadagopan N, Zheng Z, Zha H. Like like alike: joint friendship and interest propagation in social networks. In: WWW. ACM; 2011. p. 537–46.

Zhang J, Tang J, Li J. Expert finding in a social network. In: Advances in databases: concepts, systems and applications; 2007. p. 1066–9.

Zheleva E, Getoor L. Privacy in social networks: a survey. In: Social network data analytics; 2011. p. 277–306.